Министерство науки и высшего образования Российской Федерации Федеральное государственное бюджетное образовательное учреждение высшего образования «Тульский государственный университет»

КАФЕДРА ИНФОРМАЦИОННОЙ БЕЗОПАСНОСТИ

**ПРОГРАММИРОВАНИЕ АЛГОРИТМОВ ЛИНЕЙНОЙ, ВЕТВЯЩЕЙСЯ И ЦИКЛИЧЕСКОЙ СТРУКТУРЫ**

отчет о лабораторной работе № 2

по дисциплине

*ТЕХНОЛОГИИ И МЕТОДЫ ПРОГРАММИРОВАНИЯ*

***ВАРИАНТ 13***

|  |  |  |
| --- | --- | --- |
| Выполнила: | ст. гр. 230711 | Павлова В.С. |
| Проверил: | асс. каф. ИБ | Курбаков М.Ю. |

Тула, 2023 г.

**ЦЕЛЬ И ЗАДАЧА РАБОТЫ**

**Цель:** научиться использовать стандартные функции и арифметические операторы; изучить операторы отношения и присваивания, логические операции, условный оператор и оператор-переключатель; научиться использовать в программах операторы цикла.

**Задача:** в данной работе требуется написать программу, демонстрирующую использование изученных принципов.

**ЗАДАНИЕ НА РАБОТУ**

**Задание по теме «Алгоритмы линейной структуры»:**

№ 5. Вычислить периметр и площадь правильного **n**- угольника, вписанного в окружность радиуса **R** (**n** - целого типа, **R** - вещественного типа).

**Задание по теме «Ветвящиеся алгоритмы»**

№ 11. Определить сколько корней имеет уравнение квадратного трехчлена вида , причем коэффициенты a, b, c вводятся с клавиатуры.

**Задание по теме «Циклические алгоритмы»**

№ 9. Вычислить **k**-тое число последовательности целых простых чисел (где **k**>1).

# **СХЕМА ПРОГРАММЫ ДЛЯ ЗАДАНИЯ №5**

Схема алгоритма программы, предназначенной для вычисления периметра и площади правильного n-угольника, вписанного в окружность радиуса R, представлена на рисунке 1.

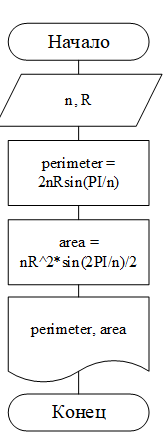


Рисунок 1 – Схема алгоритма программы, предназначенной для вычисления периметра и площади правильного n-угольника, вписанного в окружность радиуса R

# **ТЕКСТ ПРОГРАММЫ ДЛЯ ЗАДАНИЯ №5**

Текст программы на языке программирования С/С++, предназначенной для вычисления периметра и площади правильного n-угольника, вписанного в окружность радиуса R представлена в листинге 1.

## **Листинг 1. Текст программы**

#include <iostream>

#include <cmath>

# define M\_PI 3.14159265358979323846

int main()

{

setlocale(LC\_ALL, "RUSSIAN");

int n; double R;

std::cout << "Введите количество вершин n: "; std::cin >> n;

std::cout << "Введите радиус окружности R: "; std::cin >> R;

double perimeter = 2 \* n \* R \* sin(M\_PI / n);

double area = 0.5 \* n \* pow(R, 2) \* sin(2 \* M\_PI / n);

std::cout << "Периметр равен: " << perimeter << std::endl;

std::cout << "Площадь равна: " << area << std::endl;

return 0;

}

**ИНСТРУКЦИЯ ПРОГРАММИСТА ДЛЯ ЗАДАНИЯ №5**

Структуры данных, используемые в программе, приведены в таблице 1.

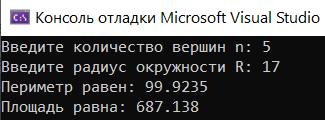
Таблица 1 – Структуры данных в программе

|  |  |  |
| --- | --- | --- |
| **Имя** | **Тип** | **Предназначение** |
| n | int | Число вершин |
| R | double | Радиус |
| perimeter | double | Периметр |
| area | double | Площадь |

# **ДЕМОНСТРАЦИОННЫЙ ПРИМЕР ДЛЯ ЗАДАНИЯ №5**

Для числа вершин n = 5 и радиуса R = 17 расчёты должны быть следующие:

Результат работы программы приведён на рисунке 2.

  
Рисунок 2 – Пример работы программы

# **СХЕМА ПРОГРАММЫ ДЛЯ ЗАДАНИЯ №11**

Схема алгоритма программы, предназначенной для определения числа корней (а также самих корней) уравнения квадратного трехчлена , где коэффициенты a, b, c вводятся с клавиатуры, представлена на рисунке 3.

  
Рисунок 3 – Схема алгоритма программы, предназначенной для определения числа корней (и самих корней) уравнения квадратного трехчлена

# **ТЕКСТ ПРОГРАММЫ ДЛЯ ЗАДАНИЯ№ 11**

Текст программы на языке программирования С/С++, предназначенной для определения числа корней (и самих корней) уравнения квадратного трехчлена представлена в листинге 2.

## **Листинг 2. Текст программы**

#include <iostream>

#include <cmath>

using namespace std;

int main()

{

setlocale(LC\_ALL, "RUSSIAN");

double a, b, c, discriminant, x1, x2;

cout << "Введите коэффициенты a, b и c: "; cin >> a >> b >> c;

discriminant = b \* b - 4 \* a \* c;

if (discriminant > 0)

{ // Два действительных корня

x1 = (-b + sqrt(discriminant)) / (2 \* a);

x2 = (-b - sqrt(discriminant)) / (2 \* a);

cout << "Уравнение имеет два действительных корня: " << x1 << " и " << x2 << "\n";

}

else if (discriminant == 0)

{ // Один действительный корень

x1 = -b / (2 \* a);

cout << "Уравнение имеет один действительный корень: " << x1 << "\n";

}

else

{ // Два комплексных корня

double realPart = -b / (2 \* a);

double imaginaryPart = sqrt(-discriminant) / (2 \* a);

cout << "Уравнение имеет два комплексных корня: " << realPart << " + "

<< imaginaryPart << "i и " << realPart << " - " << imaginaryPart << "i" << "\n";

}

return 0;

}

**ИНСТРУКЦИЯ ПРОГРАММИСТА ДЛЯ ЗАДАНИЯ №11**

Структуры данных, используемые в программе, приведены в таблице 2.

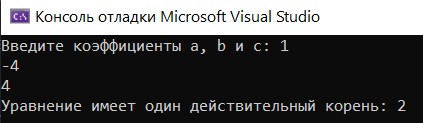
Таблица 2 – Структуры данных в программе

|  |  |  |
| --- | --- | --- |
| **Имя** | **Тип** | **Предназначение** |
| a,b,c | double | Коэффициенты |
| discriminant | double | Дискриминант |
| x1,x2 | double | Корни |
| realPart | double | Действительная часть комплексного корня |
| imaginaryPart | double | Мнимая часть комплексного корня |

# **ДЕМОНСТРАЦИОННЫЙ ПРИМЕР ДЛЯ ЗАДАНИЯ №11**

А) Для a = 1, b = -4, c = 4: ,

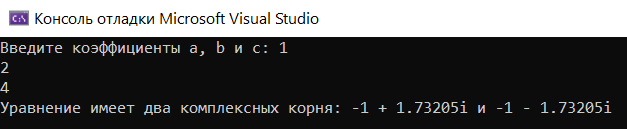
Результат работы программы приведён на рисунке 4.

  
Рисунок 4 – Пример работы программы

Б) Для a = 1, b = 2, c = 4: ,

*и*

Результат работы программы приведён на рисунке 5.

  
Рисунок 5 – Пример работы программы

# **СХЕМА ПРОГРАММЫ ДЛЯ ЗАДАНИЯ №9**

Схема алгоритма программы для вычисления **k**-го числа последовательности целых простых чисел (где **k**>1), представлена на рис. 6.

  
Рисунок 6 – Схема алгоритма программы, предназначенной для вычисления k-го числа последовательности целых простых чисел

# **ТЕКСТ ПРОГРАММЫ ДЛЯ ЗАДАНИЯ №9**

Текст программы на языке программирования С/С++, предназначенной для вычисления k-го числа последовательности целых простых чисел представлена в листинге 3.

## **Листинг 3. Текст программы**

#include <iostream>

#include <vector>

using namespace std;

bool isPrime(int n) {

if (n <= 1) {

return false;

}

for (int i = 2; i \* i <= n; i++) {

if (n % i == 0

return false;

}

return true;

}

int main() {

system("chcp 1251"); setlocale(LC\_ALL, "RUSSIAN");

int k;

cout << "Введите число k: ";

cin >> k;

vector<int> primes; // Массив для хранения простых чисел

primes.push\_back(2); cout << 2 << " "; // Первое простое число - 2

int n = 3; // Начало с 3, т.к. 2 уже учетно

while (primes.size() < k) {

if (isPrime(n)) {

primes.push\_back(n);

cout << n << " ";

}

n++;

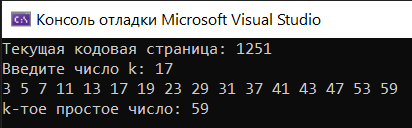
} cout << "\nk-тое простое число: " << primes.back() << endl;

return 0;

}

# **ДЕМОНСТРАЦИОННЫЙ ПРИМЕР ДЛЯ ЗАДАНИЯ №9**

Для k = 17 получим простое число 59, поскольку первые 16 простых чисел: {2 3 5 7 11 13 17 19 23 29 31 37 41 43 47 53 59}*.* Результат работы программы приведён на рисунке 7.

  
Рисунок 7 – Пример работы программы

# **ВЫВОДЫ**

В ходе выполнения работы я ознакомилась с принципами программирования алгоритмов линейной, ветвящейся и циклической структуры. Так, алгоритмом линейной структуры называется алгоритм, который состоит из последовательно выполняющихся операторов и функций. Ветвление – это алгоритмическая конструкция, в которой в зависимости от выполнения условия (да или нет) предусмотрен выбор одной из двух последовательностей команд (ветвей). Цикл — разновидность управляющей конструкции в высокоуровневых языках программирования, предназначенная для организации многократного исполнения набора инструкций